Amendments to the Claims

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claim 1 (currently amended): A single integrated circuit (IC), comprising:

a synthesizable microcontroller processor core connected to a program code memory ~~and able to~~that selectively ~~process one of two different program execution streams,~~ processes either a first program execution stream or a second program execution stream, wherein said second program execution stream is more economical with program code space than said first program execution stream, and wherein a program execution interrupt request forces a hardware switch to ~~a first of said two different program execution streams, and further wherein the execution of a particular instruction is required to switch back to a second of said two different program execution streams;~~said first program execution stream;

an interrupt controller ~~for receiving peripheral~~that receives interrupt requests ~~from a variety of system interrupt sources, and providing for~~ and provides~~said interrupt request to the core from~~ a programmable combination of said ~~peripheral~~ interrupt requests to the core, wherein each said interrupt request is associated with one of a plurality of interrupt service routines coded in said second program execution stream; and

an interrupt service routine preamble; ~~shared amongst at least two of said~~ plurality of interrupt service routines, said interrupt service routine preamble is coded in said first program execution stream to cause a hardware switch to said second program execution stream for executing an interrupt service routine, said interrupt controller causes the execution of said interrupt service routine preamble before an interrupt service routine is executed.

~~wherein, said second of said two different program execution streams is more economical with program code space, so the interrupt service routine preamble is coded in said second of said two different program execution streams to cause a switch, and is shared amongst a plurality of interrupt service routines.~~

Claim 2 (original): The IC of claim 1, wherein:

the interrupt controller provides for a fast interrupt request (FIQ) and a normal interrupt request (IRQ) to be programmably masked and prioritized before being issued

to the core, and wherein said FIQ directs program execution to a last entry in an interrupt vector table.

Claim 3 (original): The IC of claim 1, wherein:

the interrupt controller centralizes all interrupt handling and includes programmable interrupt masks for independent enabling and disabling each interrupt source, and for globally disabling all interrupts.

Claim 4 (original): The IC of claim 1, wherein:

the interrupt controller further includes an interrupt vector control for automatically decoding a highest-priority interrupt for presentation of a programmed interrupt vector to the processor core.

Claim 5 (original): The IC of claim 1, wherein:

the interrupt controller includes a global-disable control bit for use when a critical portion of program code is executing that is independent of any individual interrupt masks and avoids needing to save and restore mask states that would otherwise increase interrupt latency when a global-disable is lifted.

Claim 6: (canceled)

Claim 7 (currently amended): A single integrated circuit (IC), comprising:

a synthesizable ARM-type microcontroller processor core connected to a program code memory ~~and able to selectively process~~ that selectively processes either a THUMB or an ARM program execution stream, wherein said THUMB program execution stream is more economical with program code space, and wherein a program-execution interrupt request forces a hardware switch to said ARM program execution ~~stream, and further wherein the execution of a BX instruction is required to switch back to said THUMB program execution streams;~~

an interrupt controller ~~for receiving peripheral~~ that receives interrupt requests ~~from a variety of system interrupt sources, and providing for said interrupt request to the core from a programmable combination of said peripheral interrupt requests;~~ and provides a programmable combination of said interrupt requests to the core, wherein each said interrupt request is associated with one of a plurality of interrupt service routines coded in THUMB program code; and

an interrupt service routine preamble~~;~~ shared amongst said plurality of interrupt service routines, said interrupt service routine preamble is coded in ARM program code to cause a switch to THUMB program execution, said interrupt controller causes the

execution of said interrupt service routine preamble before an interrupt service routine is executed.

~~wherein, said THUMB program execution stream is more economical with program code space, and the interrupt service routine preamble is coded in ARM program code to cause a switch to THUMB program execution, and the interrupt service routine preamble is shared amongst a plurality of interrupt service routines to further economize on program code space.~~

Claim 8: (canceled)

Claim 9 (new): A system that includes a single integrated circuit (IC), comprising:

a synthesizable microcontroller processor core connected to a program code memory that selectively processes either a first program execution stream or a second program execution stream, wherein said second program execution stream is more economical with program code space than said first program execution stream, and wherein a program execution interrupt request forces a hardware switch to said first program-execution stream;

an interrupt controller that receives interrupt requests and provides a programmable combination of said interrupt requests to the core, wherein each said interrupt request is associated with one of a plurality of interrupt service routines coded in said second program execution stream; and

an interrupt service routine preamble shared amongst said plurality of interrupt service routines, said interrupt service routine preamble is coded in said first program execution stream to cause a hardware switch to said second program execution stream, said interrupt controller causes the execution of said interrupt service routine preamble before an interrupt service routine is executed.

Claim 10 (new): The system of claim 9, wherein:
the interrupt controller provides for a fast interrupt request (FIQ) and a normal interrupt request (IRQ) to be programmably masked and prioritized before being issued to the core, and wherein said FIQ directs program execution to a last entry in an interrupt vector table.

Claim 11 (new): The system of claim 9, wherein:
the interrupt controller centralizes all interrupt handling and includes programmable interrupt masks for independent enabling and disabling each interrupt source, and for globally disabling all interrupts.

Claim 12 (new): The system of claim 9, wherein:

the interrupt controller further includes an interrupt vector control for automatically decoding a highest-priority interrupt for presentation of a programmed interrupt vector to the processor core.

Claim 13 (new): The system of claim 9, wherein:
the interrupt controller includes a global-disable control bit for use when a critical portion of program code is executing that is independent of any individual interrupt masks and avoids needing to save and restore mask states that would otherwise increase interrupt latency when a global-disable is lifted.

Claim 14 (new): A method that makes a single integrated circuit (IC), comprising:
providing a synthesizable microcontroller processor core connected to a program code memory that selectively processes either a first program execution stream or a second program execution stream, wherein said second program execution stream is more economical with program code space than said first program execution stream, and wherein a program execution interrupt request forces a hardware switch to said first program-execution stream;

providing an interrupt controller that receives interrupt requests and provides a programmable combination of said interrupt requests to the core, wherein each said interrupt request is associated with one of a plurality of interrupt service routines coded in said second program execution stream; and

providing an interrupt service routine preamble shared amongst said plurality of interrupt service routines, said interrupt service routine preamble is coded in said first program execution stream to cause a hardware switch to said second program execution stream, said interrupt controller causes the execution of said interrupt service routine preamble before an interrupt service routine is executed.

Claim 15 (new): The method of claim 14, wherein:
the interrupt controller provides for a fast interrupt request (FIQ) and a normal interrupt request (IRQ) to be programmably masked and prioritized before being issued to the core, and wherein said FIQ directs program execution to a last entry in an interrupt vector table.

Claim 16 (new): The method of claim 14, wherein:
the interrupt controller centralizes all interrupt handling and includes programmable interrupt masks for independent enabling and disabling each interrupt source, and for globally disabling all interrupts.

Claim 17 (new): The method of claim 14, wherein:
the interrupt controller further includes an interrupt vector control for automatically decoding a highest-priority interrupt for presentation of a programmed interrupt vector to the processor core.

Claim 18 (new): The method of claim 14, wherein:
the interrupt controller includes a global-disable control bit for use when a critical portion of program code is executing that is independent of any individual interrupt masks and avoids needing to save and restore mask states that would otherwise increase interrupt latency when a global-disable is lifted.

Claim 19 (new): A method that processes interrupts using a single integrated circuit (IC), comprising:
receiving one or more program execution interrupt requests, wherein each said interrupt request causes a hardware switch in a synthesizable microcontroller processor that selectively processes either a first program execution stream or a second program execution stream that is more economical with program code space than said first program execution stream, and wherein said hardware switch forces execution to said first program-execution stream;

providing a programmable combination of said interrupt requests to the core using an interrupt controller, wherein each said interrupt request is associated with one of a plurality of interrupt service routines coded in said second program execution stream; and

executing an interrupt service routine preamble before an interrupt service routine is executed, wherein said interrupt service routine preamble is shared amongst said plurality of interrupt service routines and is coded in said first program execution stream to cause a hardware switch to said second program execution stream.

Claim 20 (new): The method of claim 19, wherein:
the interrupt controller provides for a fast interrupt request (FIQ) and a normal interrupt request (IRQ) to be programmably masked and prioritized before being issued to the core, and wherein said FIQ directs program execution to a last entry in an interrupt vector table.

Claim 21 (new): The method of claim 19, wherein:
the interrupt controller centralizes all interrupt handling and includes programmable interrupt masks for independent enabling and disabling each interrupt source, and for globally disabling all interrupts.

Claim 22 (new): The method of claim 19, wherein:
the interrupt controller further includes an interrupt vector control for automatically decoding a highest-priority interrupt for presentation of a programmed interrupt vector to the processor core.

Claim 23 (new): The method of claim 19, wherein:
the interrupt controller includes a global-disable control bit for use when a critical portion of program code is executing that is independent of any individual interrupt

masks and avoids needing to save and restore mask states that would otherwise increase interrupt latency when a global-disable is lifted.